B 10

```
PPPPPPPPPPPP     HHH         HHH     OOOOOOOOO     NNN         NNN  EEEEEEEEEEEEEEEE
PPPPPPPPPPPP     HHH         HHH     OOOOOOOOO     NNN         NNN  EEEEEEEEEEEEEEEE
PPPPPPPPPPPP     HHH         HHH     OOOOOOOOO     NNN         NNN  EEEEEEEEEEEEEEEE
PPP         PPP  HHH         HHH  OOO         OOO  NNN         NNN  EEE
PPP         PPP  HHH         HHH  OOO         OOO  NNN         NNN  EEE
PPP         PPP  HHH         HHH  OOO         OOO  NNN         NNN  EEE
PPP         PPP  HHH         HHH  OOO         OOO  NNNNNN      NNN  EEE
PPP         PPP  HHH         HHH  OOO         OOO  NNNNNN      NNN  EEE
PPP         PPP  HHH         HHH  OOO         OOO  NNNNNN      NNN  EEE
PPPPPPPPPPPP     HHHHHHHHHHHHHHH  OOO         OOO  NNN   NNN   NNN  EEEEEEEEEEEE
PPPPPPPPPPPP     HHHHHHHHHHHHHHH  OOO         OOO  NNN   NNN   NNN  EEEEEEEEEEEE
PPPPPPPPPPPP     HHHHHHHHHHHHHHH  OOO         OOO  NNN   NNN   NNN  EEEEEEEEEEEE
PPP              HHH         HHH  OOO         OOO  NNN      NNNNNN  EEE
PPP              HHH         HHH  OOO         OOO  NNN      NNNNNN  EEE
PPP              HHH         HHH  OOO         OOO  NNN      NNNNNN  EEE
PPP              HHH         HHH  OOO         OOO  NNN         NNN  EEE
PPP              HHH         HHH  OOO         OOO  NNN         NNN  EEE
PPP              HHH         HHH  OOO         OOO  NNN         NNN  EEE
PPP              HHH         HHH     OOOOOOOOO     NNN         NNN  EEEEEEEEEEEEEEEE
PPP              HHH         HHH     OOOOOOOOO     NNN         NNN  EEEEEEEEEEEEEEEE
PPP              HHH         HHH     OOOOOOOOO     NNN         NNN  EEEEEEEEEEEEEEEE
```

```
**FILE**ID**PUBSUBS
```

```
PPPPPPPP    UU      UU  BBBBBBBB      SSSSSSSS  UU      UU  BBBBBBBB      SSSSSSSS
PPPPPPPP    UU      UU  BBBBBBBB      SSSSSSSS  UU      UU  BBBBBBBB      SSSSSSSS
PP      PP  UU      UU  BB      BB  SS          UU      UU  BB      BB  SS
PP      PP  UU      UU  BB      BB  SS          UU      UU  BB      BB  SS
PP      PP  UU      UU  BB      BB  SS          UU      UU  BB      BB  SS
PP      PP  UU      UU  BB      BB  SS          UU      UU  BB      BB  SS
PPPPPPPP    UU      UU  BBBBBBBB      SSSSSS    UU      UU  BBBBBBBB      SSSSSS
PPPPPPPP    UU      UU  BBBBBBBB      SSSSSS    UU      UU  BBBBBBBB      SSSSSS
PP          UU      UU  BB      BB          SS  UU      UU  BB      BB          SS
PP          UU      UU  BB      BB          SS  UU      UU  BB      BB          SS
PP          UU      UU  BB      BB          SS  UU      UU  BB      BB          SS      ....
PP          UU      UU  BB      BB          SS  UU      UU  BB      BB          SS      ....
PP          UUUUUUUUUU  BBBBBBBB    SSSSSSSS    UUUUUUUUUU  BBBBBBBB    SSSSSSSS        ....
PP          UUUUUUUUUU  BBBBBBBB    SSSSSSSS    UUUUUUUUUU  BBBBBBBB    SSSSSSSS        ....

LL            IIIIII      SSSSSSSS
LL            IIIIII      SSSSSSSS
LL              II      SS
LL              II      SS
LL              II      SS
LL              II      SS
LL              II        SSSSSS
LL              II        SSSSSS
LL              II              SS
LL              II              SS
LL              II              SS
LL              II              SS
LLLLLLLLLL    IIIIII    SSSSSSSS
LLLLLLLLLL    IIIIII    SSSSSSSS
```

```
0 %title 'PUBSUBS - Subroutines for Handling PUBs'
0         module pubsubs (
1                         ident='V04-000') = begin
1
1 !
1 !****************************************************************************
1 !*                                                                          *
1 !*  COPYRIGHT (c) 1978, 1980, 1982, 1984 BY                                 *
1 !*  DIGITAL EQUIPMENT CORPORATION, MAYNARD, MASSACHUSETTS.                  *
1 !*  ALL RIGHTS RESERVED.                                                    *
1 !*                                                                          *
1 !*  THIS SOFTWARE IS FURNISHED UNDER A LICENSE AND MAY BE USED AND COPIED   *
1 !*  ONLY IN  ACCORDANCE WITH  THE  TERMS  OF  SUCH  LICENSE  AND WITH THE   *
1 !*  INCLUSION OF THE ABOVE COPYRIGHT NOTICE. THIS SOFTWARE OR  ANY  OTHER   *
1 !*  COPIES THEREOF MAY NOT BE PROVIDED OR OTHERWISE MADE AVAILABLE TO ANY   *
1 !*  OTHER PERSON.  NO TITLE TO AND OWNERSHIP OF  THE  SOFTWARE IS  HEREBY   *
1 !*  TRANSFERRED.                                                            *
1 !*                                                                          *
1 !*  THE INFORMATION IN THIS SOFTWARE IS  SUBJECT TO CHANGE WITHOUT NOTICE   *
1 !*  AND  SHOULD  NOT  BE  CONSTRUED AS  A COMMITMENT BY DIGITAL EQUIPMENT   *
1 !*  CORPORATION.                                                            *
1 !*                                                                          *
1 !*  DIGITAL ASSUMES NO RESPONSIBILITY FOR THE USE  OR  RELIABILITY OF ITS   *
1 !*  SOFTWARE ON EQUIPMENT WHICH IS NOT SUPPLIED BY DIGITAL.                 *
1 !*                                                                          *
1 !*                                                                          *
1 !****************************************************************************
1 !
1
1 !++
1 ! Facility:     VAX/VMS Telephone Facility, Subroutines for Handling PUBs
1 !
1 ! Abstract:     This module provides the routines necessary for handling
1 !               Phone Unit Blocks (PUB).  A PUB is used to describe the
1 !               characteristics of one of the phone units that the user
1 !               is talking to.  All information about that unit is in
1 !               its PUB.
1 !
1 !
1 ! Environment:
1 !
1 ! Author: Paul C. Anagnostopoulos, Creation Date: 18 November 1980
1 !
1 ! Modified By:
1 !
1 !       V03-001 PCA0046         Paul Anagnostopoulos    26-Mar-1982
1 !               Minor changes to convert from process name to user name.
1 !--
```

```
;    50            0049  1 %sbttl 'Module Declarations'
;    51            0050  1 !
;    52            0051  1 ! Libraries and Requires:
;    53            0052  1 !
;    54            0053  1
;    55            0054  1 library 'sys$library:starlet.l32';
;    56            0055  1 require 'phonereq';
;    57            0384  1
;    58            0385  1 !
;    59            0386  1 ! Table of Contents:
;    60            0387  1 !
;    61            0388  1
;    62            0389  1 forward routine
;    63            0390  1         phn$make_pub,
;    64            0391  1         phn$make_tsb,
;    65            0392  1         phn$redundant,
;    66            0393  1         phn$conversing,
;    67            0394  1         phn$cmp_target,
;    68            0395  1         phn$kill_pub: novalue,
;    69            0396  1         phn$make_ctl: novalue,
;    70            0397  1         phn$kill_ctl: novalue;
;    71            0398  1
;    72            0399  1 !
;    73            0400  1 ! Macro Definitions:
;    74            0401  1 !
;    75            0402  1 ! The following macro is used in the PHN$MAKE_TSB routine.
;    76            0403  1 !
;    77            0404  1
;    78          M 0405  1 macro add_token(length,pointer) =
;    79          M 0406  1 (       inc (nt[tsb_w_tkncount]);
;    80          M 0407  1         begin
;    81          M 0408  1         bind
;    82          M 0409  1                 token_dsc = nt[tsb_q_tkndsc,.nt[tsb_w_tkncount]]: descriptor;
;    83          M 0410  1
;    84          M 0411  1         token_dsc[0,0,32,0] = length;
;    85          M 0412  1         token_dsc[ptr] = .complete_spec_dsc[ptr] + .complete_spec_dsc[len];
;    86          M 0413  1         ch$move(length,pointer,.token_dsc[ptr]);
;    87          M 0414  1         complete_spec_dsc[len] = .complete_spec_dsc[len] + length;
;    88          M 0415  1         end;
;    89          M 0416  1 )
;    90            0417  1 %;
;    91            0418  1
;    92            0419  1 !
;    93            0420  1 ! External References:
;    94            0421  1 !
;    95            0422  1
;    96            0423  1 external routine
;    97            0424  1         lib$free_vm: addressing_mode(general),
;    98            0425  1         lib$get_vm: addressing_mode(general),
;    99            0426  1         lib$tparse: addressing_mode(general);
;   100            0427  1
;   101            0428  1 external literal
;   102            0429  1         lib$_syntaxerr;
;   103            0430  1
;   104            0431  1 !
;   105            0432  1 ! Own Variables:
;   106            0433  1 !
;   108         0434    1 %sbttl 'PHN$MAKE_PUB - Create New Phone Unit Block'
;   109         0435    1 !++
;   110         0436    1 ! Functional Description:
;   111         0437    1 !         This routine is called to create a new PUB.  The PUB is allocated,
;   112         0438    1 !         initialized, and chained onto the end of the PUB chain.  By chaining
;   113         0439    1 !         onto the end, we keep PUBs in the order of creation.
;   114         0440    1 !
;   115         0441    1 ! Formal Parameters:
;   116         0442    1 !         target_spec     Address of descriptor of complete node/user name spec
;   117         0443    1 !                         of person or node we need the PUB for.
;   118         0444    1 !         pub_address     Address of longword in which to return new PUB address.
;   119         0445    1 !
;   120         0446    1 ! Implicit Inputs:
;   121         0447    1 !         global data
;   122         0448    1 !
;   123         0449    1 ! Implicit Outputs:
;   124         0450    1 !         global data
;   125         0451    1 !
;   126         0452    1 ! Returned Value:
;   127         0453    1 !         phn$_badspec    Invalid target specification syntax.
;   128         0454    1 !
;   129         0455    1 ! Side Effects:
;   130         0456    1 !
;   131         0457    1 !--
;   132         0458    1
;   133         0459    1
;   134         0460    2 global routine phn$make_pub(target_spec,pub_address) = begin
;   135         0461    2
;   136         0462    2 bind
;   137         0463    2         target_spec_dsc = .target_spec: descriptor;
;   138         0464    2
;   139         0465    2 local
;   140         0466    2         status: long,
;   141         0467    2         np: ref pub,                    ! Pointer to the new PUB.
;   142         0468    2         c: ref ctl;
;   143         0469    2
;   144         0470    2
;   145         0471    2 ! We begin by allocating memory for the new PUB.
;   146         0472    2
;   147         0473    2 status = lib$get_vm(%ref(pub_k_size),np);
;   148         0474    2 check (.status);
;   149         0475    2
;   150         0476    2 ! Now we clear the PUB and fill in its length.
;   151         0477    2
;   152         0478    2 ch$fill(0,pub_k_size,.np);
;   153         0479    2 np[pub_l_length] = pub_k_size;
;   154         0480    2
;   155         0481    2 ! Now we initialize the TSB portion of the PUB with the target spec
;   156         0482    2 ! we have been passed.  If that fails, just delete the stupid thing
;   157         0483    2 ! and return an error status.
;   158         0484    2
;   159         0485    2 status = phn$make_tsb(.target_spec,np[pub_b_tsb]);
;   160         0486    3 if .status eqlu phn$_badspec then (
;   161         0487    3         phn$kill_pub(.np);
;   162         0488    3         return phn$_badspec;
;   163         0489    2 );
;   164         0490    2 check (.status);
;  165          0491  2
;  166          0492  2 ! Now we mark the PUB temporary (all new PUBs are marked as such).
;  167          0493  2
;  168          0494  2 np[pub_v_temporary] = true;
;  169          0495  2
;  170          0496  2 ! Now we allocate an initial CTL buffer with no text in it.  Thus we will
;  171          0497  2 ! be prepared to add text as the person talks.  We specify this CTL as
;  172          0498  2 ! the top of the viewport.
;  173          0499  2
;  174          0500  2 phn$make_ctl(0,c);
;  175          0501  2 np[pub_q_ctlhead0] = np[pub_q_ctlhead1] = np[pub_q_ctlhead0];
;  176          0502  2 insque(.c,np[pub_q_ctlhead0]);
;  177          0503  2 inc (np[pub_l_ctlcount]);
;  178          0504  2 np[pub_l_topctl] = .np[pub_l_ctlcount];
;  179          0505  2
;  180          0506  2 ! Finally, we queue the PUB at the end of the PUB chain and return its
;  181          0507  2 ! address as requested.
;  182          0508  2
;  183          0509  2 insque(.np,.phn$gq_pubhead[1]);
;  184          0510  2 .pub_address = .np;
;  185          0511  2 return phn$_ok;
;  186          0512  2
;  187          0513  1 end;

                                                                  .TITLE  PUBSUBS PUBSUBS - Subroutines for Handling PUBs
                                                                  .IDENT  \V04-000\

                                                                  .EXTRN  PHN$_OK, PHN$_ANSWERED
                                                                  .EXTRN  PHN$_BUSYCALL, PHN$_CANCALL
                                                                  .EXTRN  PHN$_CANTREACH, PHN$_CONFCALL
                                                                  .EXTRN  PHN$_DEAD, PHN$_DECNETLINK
                                                                  .EXTRN  PHN$_DIRCAN, PHN$_FACSCAN
                                                                  .EXTRN  PHN$_HELPCAN, PHN$_HUNGUP
                                                                  .EXTRN  PHN$_JUSTRANG, PHN$_LOGGEDOFF
                                                                  .EXTRN  PHN$_REJECTED, PHN$_RING
                                                                  .EXTRN  PHN$_REJECTJUNK
                                                                  .EXTRN  PHN$_SENDINGMAIL
                                                                  .EXTRN  PHN$_BADCMD, PHN$_BADHELP
                                                                  .EXTRN  PHN$_BADMAILCMD
                                                                  .EXTRN  PHN$_BADSMB, PHN$_BADSPEC
                                                                  .EXTRN  PHN$_HELPMISSING
                                                                  .EXTRN  PHN$_IVREDUNANS
                                                                  .EXTRN  PHN$_IVREDUNCALL
                                                                  .EXTRN  PHN$_LINKERROR, PHN$_NEEDUSER
                                                                  .EXTRN  PHN$_NOCALL, PHN$_NOHOLDS
                                                                  .EXTRN  PHN$_NOPORTS, PHN$_NOPRIV
                                                                  .EXTRN  PHN$_NOPROC, PHN$_NOTCONV
                                                                  .EXTRN  PHN$_ONLYNODE, PHN$_PHONEBUSY
                                                                  .EXTRN  PHN$_REMOTEERROR
                                                                  .EXTRN  PHN$_TARGTERM, PHN$_UNPLUGGED
                                                                  .EXTRN  PHN$_BADTERM, PHN$_SHAREDMBX
                                                                  .EXTRN  PHN$_INPUTTERM, PHN$GQ_NODE_NAME
                                                                  .EXTRN  PHN$GQ_SWITCH_HOOK
                                                                  .EXTRN  PHN$GL_VIEWPORT_SIZE
                                                                  .EXTRN  PHN$GB_SCROLL, PHN$GQ_PUBHEAD
                                                                  .EXTRN  PHN$GB_FLAGS, LIB$FREE_VM
```

```
                                                                            .EXTRN   LIB$GET_VM, LIB$TPARSE
                                                                            .EXTRN   LIB$_SYNTAXERR

                                                                            .PSECT   $CODE$,NOWRT,2

                                                        03FC 00000          .ENTRY   PHN$MAKE_PUB, Save R2,R3,R4,R5,R6,R7,R8,R9    ; 0460
                                         59 00000000G 8F  D0 00002          MOVL     #PHN$_BADSPEC, R9                             ;
                                         58 00000000G 00  9E 00009          MOVAB    LIB$SIGNAL, R8                                ;
                                         5E           0C  C2 00010          SUBL2    #12, SP                                       ;
                                                  04  AE  9F 00013          PUSHAB   NP                                            ; 0473
                                   04    AE     0110  8F  3C 00016          MOVZWL   #272, 4(SP)                                   ;
                                                  04  AE  9F 0001C          PUSHAB   4(SP)                                         ;
                             00000000G   00           02  FB 0001F          CALLS    #2, LIB$GET_VM                                ;
                                         57           50  D0 00026          MOVL     R0, STATUS                                    ;
                                         05           57  E8 00029          BLBS     STATUS, 1$                                    ; 0474
                                                      57  DD 0002C          PUSHL    STATUS                                        ;
                                         68           01  FB 0002E          CALLS    #1, LIB$SIGNAL                                ;
                                         56       04  AE  D0 00031 1$:      MOVL     NP, R6                                        ; 0478
    0110    8F            00             6E           00  2C 00035          MOVC5    #0, (SP), #0, #272, (R6)                      ;
                                                      66     0003C                                                                 ;
                                   08    A6     0110  8F  3C 0003D          MOVZWL   #272, 8(R6)                                   ; 0479
                                                  0C  A6  9F 00043          PUSHAB   12(R6)                                        ; 0485
                                                  04  AC  DD 00046          PUSHL    TARGET_SPEC                                   ;
                                 0000V   CF           02  FB 00049          CALLS    #2, PHN$MAKE_TSB                              ;
                                         57           50  D0 0004E          MOVL     R0, STATUS                                    ;
                                         59           57  D1 00051          CMPL     STATUS, R9                                    ; 0486
                                                      0B  12 00054          BNEQ     2$                                            ;
                                                      56  DD 00056          PUSHL    R6                                            ; 0487
                                 0000V   CF           01  FB 00058          CALLS    #1, PHN$KILL_PUB                              ;
                                         50           59  D0 0005D          MOVL     R9, R0                                        ; 0488
                                                          04 00060          RET                                                    ;
                                         05           57  E8 00061 2$:      BLBS     STATUS, 3$                                    ; 0490
                                                      57  DD 00064          PUSHL    STATUS                                        ;
                                         68           01  FB 00066          CALLS    #1, LIB$SIGNAL                                ;
                                 00F0    C6           04  88 00069 3$:      BISB2    #4, 240(R6)                                   ; 0494
                                                  08  AE  9F 0006E          PUSHAB   C                                             ; 0500
                                                      7E  D4 00071          CLRL     -(SP)                                         ;
                                 0000V   CF           02  FB 00073          CALLS    #2, PHN$MAKE_CTL                              ;
                                         50     0104  C6  9E 00078          MOVAB    260(R6), R0                                   ; 0501
                                 0108    C6           50  D0 0007D          MOVL     R0, 264(R6)                                   ;
                                         60           50  D0 00082          MOVL     R0, (R0)                                      ;
                                         60       08  BE  0E 00085          INSQUE   @C, (R0)                                      ; 0502
                                         50       04  AE  D0 00089          MOVL     NP, R0                                        ; 0503
                                                0100  C0  D6 0008D          INCL     256(R0)                                       ;
                                 010C    C0     0100  C0  D0 00091          MOVL     256(R0), 268(R0)                              ; 0504
                                 0000G   DF           60  0E 00098          INSQUE   (R0), @PHN$GQ_PUBHEAD+4                       ; 0509
                                   08    BC       04  AE  D0 0009D          MOVL     NP, @PUB_ADDRESS                              ; 0510
                                         50 00000000G 8F  D0 000A2          MOVL     #PHN$_OK, R0                                  ; 0511
                                                          04 000A9          RET                                                    ; 0513

; Routine Size:  170 bytes,    Routine Base:  $CODE$ + 0000
;  189        0514  1 %sbttl 'PHN$MAKE_TSB - Build a Target Specification Block'
;  190        0515  1 !++
;  191        0516  1 ! Functional Description:
;  192        0517  1 !       This routine is called to construct a Target Specification Block,
;  193        0518  1 !       a data structure containing the complete node and user name
;  194        0519  1 !       specification for some person or node we want to communicate with.
;  195        0520  1 !
;  196        0521  1 !       The original version of this routine used TPARSE, but it
;  197        0522  1 !       was rewritten by hand because of its frequency of use.
;  198        0523  1 !
;  199        0524  1 ! Formal Parameters:
;  200        0525  1 !       target_spec     The node/user string in any valid format.
;  201        0526  1 !       new_tsb         The address of the TSB to be filled in.
;  202        0527  1 !
;  203        0528  1 ! Implicit Inputs:
;  204        0529  1 !       global data
;  205        0530  1 !
;  206        0531  1 ! Implicit Outputs:
;  207        0532  1 !       global data
;  208        0533  1 !
;  209        0534  1 ! Returned Value:
;  210        0535  1 !       phn$_badspec    The target spec was syntactically incorrect.
;  211        0536  1 !                       Currently this is never the case.
;  212        0537  1 !
;  213        0538  1 ! Side Effects:
;  214        0539  1 !
;  215        0540  1 !--
;  216        0541  1
;  217        0542  1
;  218        0543  2 global routine phn$make_tsb(target_spec,new_tsb) = begin
;  219        0544  2
;  220        0545  2 bind
;  221        0546  2         target_spec_dsc = .target_spec: descriptor,
;  222        0547  2         nt = .new_tsb: tsb,
;  223        0548  2         complete_spec_dsc = nt[tsb_q_tkndsc,0]: descriptor;
;  224        0549  2
;  225        0550  2 local
;  226        0551  2         status: long,
;  227        0552  2         spec_ptr: long, spec_ptr2: long,
;  228        0553  2         spec_end_ptr: long;
;  230     0554  2 ! We begin by clearing the new TSB and initializing the complete spec
;  231     0555  2 ! descriptor.
;  232     0556  2
;  233     0557  2 ch$fill(0,tsb_k_size,nt);
;  234     0558  2 complete_spec_dsc[ptr] = nt[tsb_t_string];
;  235     0559  2
;  236     0560  2 ! Now we look at the target spec and see if it contains a node name.
;  237     0561  2
;  238     0562  2 spec_ptr = .target_spec_dsc[ptr];
;  239     0563  2 spec_end_ptr = .spec_ptr + .target_spec_dsc[len];
;  240     0564  2
;  241     0565  2 if ch$find_sub(.target_spec_dsc[len],.target_spec_dsc[ptr],
;  242     0566  2               2,uplit byte('::')) eqla 0                          then
;  243     0567  2
;  244     0568  2         ! The spec does not contain a node name.  We will therefore use
;  245     0569  2         ! our node name as the default, unless we are not on a DECnet node.
;  246     0570  2
;  247     0571  2         if .phn$gq_node_name[len] nequ 0 then
;  248     0572  3                 add_token(.phn$gq_node_name[len],.phn$gq_node_name[ptr])
;  249     0573  2         else
;  250     0574  3                 add_token(0,0)
;  251     0575  3
;  252     0576  2 else
;  253     0577  2         ! The target spec contains at least one node name.  Now we have
;  254     0578  2         ! to scan off the node names and add them to the spec in the TSB.
;  255     0579  2         ! We normalize then by removing blanks and leading underscores.
;  256     0580  2         ! We also add build a descriptor for each one.
;  257     0581  2
;  258     0582  3         loop (
;  259     0583  3                 spec_ptr = ch$find_not_ch(.spec_end_ptr-.spec_ptr,
;  260     0584  3                                         .spec_ptr,' ');
;  261     0585  3                 if .spec_ptr eqla 0 then
;  262     0586  3                         spec_ptr = .spec_end_ptr;
;  263     0587  3                 spec_ptr2 = ch$find_sub(.spec_end_ptr-.spec_ptr,.spec_ptr,
;  264     0588  3                                         2,uplit byte('::'));
;  265     0589  3         exitif (.spec_ptr2 eqla 0);
;  266     0590  3
;  267     0591  3                 if ch$rchar(.spec_ptr) eqlu '_' then
;  268     0592  3                         inc (spec_ptr);
;  269     0593  3                 add_token(.spec_ptr2-.spec_ptr+2,.spec_ptr);
;  270     0594  3                 spec_ptr = .spec_ptr2 + 2;
;  271     0595  2         );
;  272     0596  2
;  273     0597  2 ! We have finished processing the node names.  Now we have to isolate the
;  274     0598  2 ! user name (if any), add it to the spec in the TSB, and build a descriptor.
;  275     0599  2
;  276     0600  2 spec_ptr = ch$find_not_ch(.spec_end_ptr-.spec_ptr, .spec_ptr,' ');
;  277     0601  2 if .spec_ptr eqla 0 then
;  278     0602  2         spec_ptr = .spec_end_ptr;
;  279     0603  2 add_token(.spec_end_ptr-.spec_ptr,.spec_ptr);
;   281             0604  2 ! If the explicit or default home node on the spec is not our node, then
;   282             0605  2 ! we need to set the remote flag in the TSB.
;   283             0606  2
;   284             0607  3 begin
;   285             0608  3 bind
;   286             0609  3         home_node_dsc = nt[tsb_q_tkndsc,.nt[tsb_w_tkncount]-1]: descriptor;
;   287             0610  3
;   288             0611  3 nt[tsb_v_remote] = ch$neq(.home_node_dsc[len],.home_node_dsc[ptr],
;   289             0612  3                           .phn$gq_node_name[len],.phn$gq_node_name[ptr],' ');
;   290             0613  2 end;
;   291             0614  2
;   292             0615  2 ! If a user name was specified in the spec, we need to set the user
;   293             0616  2 ! flag in the TSB.  If none was specified, the length of the user name
;   294             0617  2 ! token was set to zero.
;   295             0618  2
;   296             0619  3 begin
;   297             0620  3 bind
;   298             0621  3         user_name_dsc = nt[tsb_q_tkndsc,.nt[tsb_w_tkncount]]: descriptor;
;   299             0622  3
;   300             0623  3 nt[tsb_v_user] = .user_name_dsc[len] nequ 0;
;   301             0624  2 end;
;   302             0625  2
;   303             0626  2 return phn$_ok;
;   304             0627  2
;   305             0628  1 end;



                                                                        .PSECT  $PLIT$,NOWRT,NOEXE,2

                                              3A  3A  00000 P.AAA:      .ASCII  \::\                                     ;
                                              3A  3A  00002 P.AAB:      .ASCII  \::\


                                                                        .PSECT  $CODE$,NOWRT,2

                                                  0FFC  00000           .ENTRY  PHN$MAKE_TSB, Save R2,R3,R4,R5,R6,R7,R8,R9,-; 0543
                                                                                R10,R11                                    :
                            5E              04  C2  00002               SUBL2   #4, SP                                     :
                            58        04    AC  D0  00005               MOVL    TARGET_SPEC, R8                            ; 0546
                            57        08    AC  D0  00009               MOVL    NEW_TSB, R7                                ; 0547
                            59        04    A7  9E  0000D               MOVAB   4(R7), R9                                  ; 0548
   00E4   8F        00      6E              00  2C  00011               MOVC5   #0, (SP), #0, #228, (R7)                   ; 0557
                                            67      00018                                                              :
                            5B        04    A9  9E  00019               MOVAB   4(R9), R11                                 ; 0558
                            6B        54    A7  9E  0001D               MOVAB   84(R7), (R11)                              :
                            5A        04    A8  D0  00021               MOVL    4(R8), SPEC_PTR                            ; 0562
                            56              68  3C  00025               MOVZWL  (R8), SPEC_END_PTR                         ; 0563
                            56              5A  C0  00028               ADDL2   SPEC_PTR, SPEC_END_PTR                     :
     04   B8        68   0000'  CF          02  39  0002B               MATCHC  #2, P.AAA, (R8), @4(R8)                    ; 0566
                                            03  13  00033               BEQL    1$                                         :
                            53              02  D0  00035               MOVL    #2, R3                                     :
                            53              02  C2  00038 1$:           SUBL2   #2, R3                                     :
                                            48  12  0003B               BNEQ    4$                                         :
                            58     0000G    CF  3C  0003D               MOVZWL  PHN$GQ_NODE_NAME, R8                       ; 0571
                                            23  13  00042               BEQL    2$                                         :
                                                    02   A7  B6 00044        INCW    2(R7)                               ; 0572
                                           50       02   A7  3C 00047        MOVZWL  2(R7), R0                           :
                                           50       04 A740  7E 0004B        MOVAQ   4(R7)[R0], R0                       :
                                           60            58  D0 00050        MOVL    R8, (R0)                            :
                                           51            69  3C 00053        MOVZWL  (R9), R1                            :
                        04   A0            6B            51  C1 00056        ADDL3   R1, (R11), 4(R0)                    :
                        04   B0     0000G  DF            58  28 0005B        MOVC3   R8, @PHN$GQ_NODE_NAME+4, @4(R0)     :
                                           69            58  A0 00062        ADDW2   R8, (R9)                            :
                                                         1C  11 00065        BRB     3$                                  ; 0571
                                                    02   A7  B6 00067 2$:    INCW    2(R7)                               ; 0574
                                           50       02   A7  3C 0006A        MOVZWL  2(R7), R0                           :
                                           50       04 A740  7E 0006E        MOVAQ   4(R7)[R0], R0                       :
                                           60            D4  00073           CLRL    (R0)                                :
                                           51            69  3C 00075        MOVZWL  (R9), R1                            :
                        04   A0            6B            51  C1 00078        ADDL3   R1, (R11), 4(R0)                    :
          04    B0           00            00            00  F0 0007D        INSV    #0, #0, #0, @4(R0)                  :
                                                         65  11 00083 3$:    BRB     9$                                  ; 0571
                             50            56            5A  C3 00085 4$:    SUBL3   SPEC_PTR, SPEC_END_PTR, R0          ; 0583
                             6A            50            20  3B 00089        SKPC    #32, R0, (SPEC_PTR)                 :
                                                         02  12 0008D        BNEQ    5$                                  :
                                                         51  D4 0008F        CLRL    R1                                  :
                                           5A            51  D0 00091 5$:    MOVL    R1, SPEC_PTR                        :
                                                         03  12 00094        BNEQ    6$                                  ; 0585
                                           5A            56  D0 00096        MOVL    SPEC_END_PTR, SPEC_PTR              ; 0586
                             50            56            5A  C3 00099 6$:    SUBL3   SPEC_PTR, SPEC_END_PTR, R0          ; 0587
                6A           50     0000'  CF            02  39 0009D        MATCHC  #2, P.AAB, R0, (SPEC_PTR)           ; 0588
                                                         03  13 000A4        BEQL    7$                                  :
                                           53            02  D0 000A6        MOVL    #2, R3                              :
                                           6E            73  3E 000A9 7$:    MOVAW   -(R3), SPEC_PTR2                    :
                                                         3C  13 000AC        BEQL    9$                                  ; 0589
                                       5F  8F            6A  91 000AE        CMPB    (SPEC_PTR), #95                     ; 0591
                                                         02  12 000B2        BNEQ    8$                                  :
                                                         5A  D6 000B4        INCL    SPEC_PTR                            ; 0592
                                                    02   A7  B6 000B6 8$:    INCW    2(R7)                               ; 0593
                                           50       02   A7  3C 000B9        MOVZWL  2(R7), R0                           :
                                           50       04 A740  7E 000BD        MOVAQ   4(R7)[R0], R0                       :
                             53            6E            5A  C3 000C2        SUBL3   SPEC_PTR, SPEC_PTR2, R3             :
                                           53            02  C0 000C6        ADDL2   #2, R3                              :
                                           60            53  D0 000C9        MOVL    R3, (R0)                            :
                                           58            69  3C 000CC        MOVZWL  (R9), R8                            :
                        04   A0            6B            58  C1 000CF        ADDL3   R8, (R11), 4(R0)                    :
                        04   B0            6A            53  28 000D4        MOVC3   R3, (SPEC_PTR), @4(R0)              :
                             50            58            6E  C1 000D9        ADDL3   SPEC_PTR2, R8, R0                   :
                                           50            5A  C2 000DD        SUBL2   SPEC_PTR, R0                        :
                             69            50            02  A1 000E0        ADDW3   #2, R0, (R9)                        :
                             5A            6E            02  C1 000E4        ADDL3   #2, SPEC_PTR2, SPEC_PTR             ; 0594
                                                         9B  11 000E8        BRB     4$                                  ; 0576
                             50            56            5A  C3 000EA 9$:    SUBL3   SPEC_PTR, SPEC_END_PTR, R0          ; 0600
                             6A            50            20  3B 000EE        SKPC    #32, R0, (SPEC_PTR)                 :
                                                         02  12 000F2        BNEQ    10$                                 :
                                                         51  D4 000F4        CLRL    R1                                  :
                                           5A            51  D0 000F6 10$:   MOVL    R1, SPEC_PTR                        :
                                                         03  12 000F9        BNEQ    11$                                 ; 0601
                                           5A            56  D0 000FB        MOVL    SPEC_END_PTR, SPEC_PTR              ; 0602
                                                    02   A7  B6 000FE 11$:   INCW    2(R7)                               ; 0603
                                           50       02   A7  3C 00101        MOVZWL  2(R7), R0                           :
                                           50       04 A740  7E 00105        MOVAQ   4(R7)[R0], R0                       :
```

```
                   51         56           5A  C3 0010A       SUBL3   SPEC_PTR, SPEC_END_PTR, R1              ;
                              60           51  D0 0010E       MOVL    R1, (R0)                                ;
                              58           69  3C 00111       MOVZWL  (R9), R8                                ;
               04  A0         6B           58  C1 00114       ADDL3   R8, (R11), 4(R0)                        ;
               04  B0         6A           51  28 00119       MOVC3   R1, (SPEC_PTR), @4(R0)                  ;
                   50         58           56  C1 0011E       ADDL3   SPEC_END_PTR, R8, R0                    ;
                   69         50           5A  A3 00122       SUBW3   SPEC_PTR, R0, (R9)                      ;
                              50      02   A7  3C 00126       MOVZWL  2(R7), R0                               ; 0609
                              50      FC A740  7E 0012A       MOVAQ   -4(R7)[R0], R0                          ;
                                               54  D4 0012F       CLRL    R4                                      ; 0611
     0000G  CF     20     04  B0           60  2D 00131       CMPC5   (R0), @4(R0), #32, PHN$GQ_NODE_NAME, -  ;
                                  0000G    DF     00139               @PHN$GQ_NODE_NAME+4                     ;
                                               02  13 0013C       BEQL    12$                                     ;
                                               54  D6 0013E       INCL    R4                                      ;
            67     01         00           54  F0 00140 12$:  INSV    R4, #0, #1, (R7)                        ;
                              50      02   A7  3C 00145       MOVZWL  2(R7), R0                               ; 0621
                                               51  D4 00149       CLRL    R1                                      ; 0623
                                      04 A740  7F 0014B       PUSHAQ  4(R7)[R0]                               ;
                                               9E  B5 0014F       TSTW    @(SP)+                                  ;
                                               02  13 00151       BEQL    13$                                     ;
                                               51  D6 00153       INCL    R1                                      ;
            67     01         01           51  F0 00155 13$:  INSV    R1, #1, #1, (R7)                        ;
                              50 00000000G 8F  D0 0015A       MOVL    #PHN$_OK, R0                            ; 0626
                                                   04 00161       RET                                             ; 0628

; Routine Size:  354 bytes,    Routine Base:  $CODE$ + 00AA
;   307         0629  1 %sbttl 'PHN$REDUNDANT - Check for Redundant Link'
;   308         0630  1 !++
;   309         0631  1 ! Functional Description:
;   310         0632  1 !       This routine is called to check for a redundant link, that is,
;   311         0633  1 !       a link with a person who is already linked to.  The PUB being
;   312         0634  1 !       checked is assumed to still be temporary!
;   313         0635  1 !
;   314         0636  1 ! Formal Parameters:
;   315         0637  1 !       the_pub         Address of PUB for link that might be redundant.
;   316         0638  1 !
;   317         0639  1 ! Implicit Inputs:
;   318         0640  1 !       global data
;   319         0641  1 !
;   320         0642  1 ! Implicit Outputs:
;   321         0643  1 !       global data
;   322         0644  1 !
;   323         0645  1 ! Returned Value:
;   324         0646  1 !       true if link is redundant, false otherwise.
;   325         0647  1 !
;   326         0648  1 ! Side Effects:
;   327         0649  1 !
;   328         0650  1 !--
;   329         0651  1
;   330         0652  1
;   331         0653  2 global routine phn$redundant(the_pub) = begin
;   332         0654  2
;   333         0655  2 bind
;   334         0656  2         tp = .the_pub: pub;
;   335         0657  2
;   336         0658  2 local
;   337         0659  2         p: ref pub;
;   338         0660  2
;   339         0661  2
;   340         0662  2 ! Basically, we just search all non-temporary PUBs for one with the same
;   341         0663  2 ! target.  We don't search ours, however, because we do want to allow
;   342         0664  2 ! the user to call him/herself once.
;   343         0665  2
;   344         0666  2 p = ..phn$gq_pubhead[0];
;   345         0667  3 until .p eqla phn$gq_pubhead do (
;   346         0668  3         if not .p[pub_v_temporary] and
;   347         0669  3            phn$cmp_target(p[pub_b_tsb],tp[pub_b_tsb]) then
;   348         0670  3                 return true;
;   349         0671  3
;   350         0672  3         p = .p[pub_l_flink];
;   351         0673  2 );
;   352         0674  2
;   353         0675  2 return false;
;   354         0676  2
;   355         0677  1 end;



                                       0004 00000           .ENTRY  PHN$REDUNDANT, Save R2                         ; 0653
                       52     0000G  DF  D0 00002           MOVL    @PHN$GQ_PUBHEAD, P                             ; 0666
                       50     0000G  CF  9E 00007 1$:       MOVAB   PHN$GQ_PUBHEAD, R0                             ; 0667
```

```
                           50              52 D1 0000C           CMPL    P, R0                              ;
                                           1F 13 0000F           BEQL    3$                                 ;
              14     00F0  C2              02 E0 00011           BBS     #2, 240(P), 2$                     ; 0668
              7E       04  AC              0C C1 00017           ADDL3   #12, THE_PUB, -(SP)                ; 0669
                                     0C    A2 9F 0001C           PUSHAB  12(P)                              ;
                    0000V  CF              02 FB 0001F           CALLS   #2, PHN$CMP_TARGET                 ;
                           04              50 E9 00024           BLBC    R0, 2$                             ;
                           50              01 D0 00027           MOVL    #1, R0                             ; 0670
                                              04 0002A           RET                                        ;
                           52              62 D0 0002B 2$:       MOVL    (P), P                             ; 0672
                                           D7 11 0002E           BRB     1$                                 ; 0667
                                           50 D4 00030 3$:       CLRL    R0                                 ; 0675
                                              04 00032           RET                                        ; 0677

; Routine Size:  51 bytes,    Routine Base:  $CODE$ + 020C
;   357      0678   1 %sbttl 'PHN$CONVERSING - Is Conversation in Progress?'
;   358      0679   1 !++
;   359      0680   1 ! Functional Description:
;   360      0681   1 !       This routine is called to find out if there is a conversation
;   361      0682   1 !       currently in progress.  Such is the case unless every PUB is
;   362      0683   1 !       either temporary, we have on hold, or has us on hold.
;   363      0684   1 !
;   364      0685   1 ! Formal Parameters:
;   365      0686   1 !       none
;   366      0687   1 !
;   367      0688   1 ! Implicit Inputs:
;   368      0689   1 !       global data
;   369      0690   1 !
;   370      0691   1 ! Implicit Outputs:
;   371      0692   1 !       global data
;   372      0693   1 !
;   373      0694   1 ! Returned Value:
;   374      0695   1 !       1 if conversation in progress, 0 otherwise.
;   375      0696   1 !
;   376      0697   1 ! Side Effects:
;   377      0698   1 !
;   378      0699   1 !--
;   379      0700   1
;   380      0701   1
;   381      0702   2 global routine phn$conversing = begin
;   382      0703   2
;   383      0704   2 local
;   384      0705   2         p: ref pub;
;   385      0706   2
;   386      0707   2
;   387      0708   2 ! Basically, we just search every PUB (but not our own) to see if
;   388      0709   2 ! any of them are in conversation with us.
;   389      0710   2
;   390      0711   2 p = ..phn$gq_pubhead[0];
;   391      0712   3 until .p eqla phn$gq_pubhead do (
;   392      0713   3         if not .p[pub_v_temporary] and
;   393      0714   3            not .p[pub_v_uhaveheld] and
;   394      0715   3            not .p[pub_v_hasuheld]         then
;   395      0716   3                 return true;
;   396      0717   3
;   397      0718   3         p = .p[pub_l_flink];
;   398      0719   2 );
;   399      0720   2
;   400      0721   2 return false;
;   401      0722   2
;   402      0723   1 end;
```

```
                               0000 00000          .ENTRY  PHN$CONVERSING, Save nothing                    ; 0702
             51      0000G  DF   D0 00002          MOVL    @PHN$GQ_PUBHEAD, P                              ; 0711
             50      0000G  CF   9E 00007 1$:      MOVAB   PHN$GQ_PUBHEAD, R0                              ; 0712
             50             51   D1 0000C          CMPL    P, R0                                           ;
                            19   13 0000F          BEQL    3$                                              ;
             50      00F0   C1   9E 00011          MOVAB   240(P), R0                                      ; 0713
```

```
                    0B          60          02  E0 00016         BBS     #2, (R0), 2$                         ;
                                08          60  E8 0001A         BLBS    (R0), 2$                             ; 0714
                    04          60          01  E0 0001D         BBS     #1, (R0), 2$                         ; 0715
                                50          01  D0 00021         MOVL    #1, R0                               ; 0716
                                                04 00024         RET                                          ;
                                51          61  D0 00025 2$:     MOVL    (P), P                               ; 0718
                                            DD  11 00028         BRB     1$                                   ; 0712
                                            50  D4 0002A 3$:     CLRL    R0                                   ; 0721
                                                04 0002C         RET                                          ; 0723

; Routine Size:  45 bytes,    Routine Base:  $CODE$ + 023F
;   404          0724  1 %sbttl 'PHN$CMP_TARGET - Compare TSB Targets'
;   405          0725  1 !++
;   406          0726  1 ! Functional Description:
;   407          0727  1 !       This routine is called to compare the targets specs in two TSBs.
;   408          0728  1 !       The portion compared consists of the home node and the user name.
;   409          0729  1 !
;   410          0730  1 ! Formal Parameters:
;   411          0731  1 !       tsb1,tsb2       The addresses of the TSBs to be compared.
;   412          0732  1 !
;   413          0733  1 ! Implicit Inputs:
;   414          0734  1 !       global data
;   415          0735  1 !
;   416          0736  1 ! Implicit Outputs:
;   417          0737  1 !       global data
;   418          0738  1 !
;   419          0739  1 ! Returned Value:
;   420          0740  1 !       true if targets are equal, false otherwise.
;   421          0741  1 !
;   422          0742  1 ! Side Effects:
;   423          0743  1 !
;   424          0744  1 !--
;   425          0745  1
;   426          0746  1
;   427          0747  2 global routine phn$cmp_target(tsb1,tsb2) = begin
;   428          0748  2
;   429          0749  2 bind
;   430          0750  2         the_tsb1 = .tsb1: tsb,
;   431          0751  2         the_tsb2 = .tsb2: tsb,
;   432          0752  2
;   433          0753  2         home_node1_dsc = the_tsb1[tsb_q_tkndsc,.the_tsb1[tsb_w_tkncount]-1]: descriptor,
;   434          0754  2         user_name1_dsc = the_tsb1[tsb_q_tkndsc,.the_tsb1[tsb_w_tkncount]]: descriptor,
;   435          0755  2
;   436          0756  2         home_node2_dsc = the_tsb2[tsb_q_tkndsc,.the_tsb2[tsb_w_tkncount]-1]: descriptor,
;   437          0757  2         user_name2_dsc = the_tsb2[tsb_q_tkndsc,.the_tsb2[tsb_w_tkncount]]: descriptor;
;   438          0758  2
;   439          0759  2 return
;   440          0760  2     ch$eql(.home_node1_dsc[len]+.user_name1_dsc[len],.home_node1_dsc[ptr],
;   441          0761  2             .home_node2_dsc[len]+.user_name2_dsc[len],.home_node2_dsc[ptr],
;   442          0762  2             ' ');
;   443          0763  2
;   444          0764  1 end;



                               007C 00000          .ENTRY  PHN$CMP TARGET, Save R2,R3,R4,R5,R6                ; 0747
                 54    04   AC   D0 00002          MOVL    TSB1, R4                                           ; 0750
                 53    08   AC   D0 00006          MOVL    TSB2, R3                                           ; 0751
                 51    02   A4   3C 0000A          MOVZWL  2(R4), R1                                          ; 0753
                 55    FC A441   7E 0000E          MOVAQ   -4(R4)[R1], R5                                     ;
                 50    02   A3   3C 00013          MOVZWL  2(R3), R0                                          ; 0756
                 52    FC A340   7E 00017          MOVAQ   -4(R3)[R0], R2                                     ;
                 56         65   3C 0001C          MOVZWL  (R5), R6                                           ; 0760
                       04 A441   7F 0001F          PUSHAQ  4(R4)[R1]                                          ;
                 51         9E   3C 00023          MOVZWL  @(SP)+, R1                                         ;
                 51         56   C0 00026          ADDL2   R6, R1                                             ;
```

```
                          54            62  3C 00029           MOVZWL   (R2), R4                                ; 0761
                                  04 A340  7F 0002C           PUSHAQ   4(R3)[R0]                               ;
                          50            9E  3C 00030           MOVZWL   @(SP)+, R0                              ;
                          50            54  C0 00033           ADDL2    R4, R0                                  ;
                                        54  D4 00036           CLRL     R4                                      ; 0760
         50          20       04  B5    51  2D 00038           CMPC5    R1, @4(R5), #32, R0, @4(R2)             ;
                                  04    B2     0003E                                                            ;
                                        02  12 00040           BNEQ     1$                                      ;
                                        54  D6 00042           INCL     R4                                      ;
                          50            54  D0 00044 1$:       MOVL     R4, R0                                  ;
                                            04 00047           RET                                              ; 0764

; Routine Size:  72 bytes,    Routine Base:  $CODE$ + 026C
;  446         0765  1 %sbttl 'PHN$KILL_PUB - Kill Obsolete PUB'
;  447         0766  1 !++
;  448         0767  1 ! Functional Description:
;  449         0768  1 !        This routine is called to deallocate an obsolete PUB.  We unhook
;  450         0769  1 !        it from the PUB chain, deallocate any CTL buffers hanging off it,
;  451         0770  1 !        and free it up.
;  452         0771  1 !
;  453         0772  1 ! Formal Parameters:
;  454         0773  1 !        obsolete_pub     The address of the obsolete PUB.  It is assumed to
;  455         0774  1 !                         still be in the PUB chain.
;  456         0775  1 !
;  457         0776  1 ! Implicit Inputs:
;  458         0777  1 !        global data
;  459         0778  1 !
;  460         0779  1 ! Implicit Outputs:
;  461         0780  1 !        global data
;  462         0781  1 !
;  463         0782  1 ! Returned Value:
;  464         0783  1 !        none
;  465         0784  1 !
;  466         0785  1 ! Side Effects:
;  467         0786  1 !
;  468         0787  1 !--
;  469         0788  1
;  470         0789  1
;  471         0790  2 global routine phn$kill_pub(obsolete_pub): novalue = begin
;  472         0791  2
;  473         0792  2 local
;  474         0793  2         status: long,
;  475         0794  2         p: ref pub,
;  476         0795  2         c: ref ctl;
;  477         0796  2
;  478         0797  2
;  479         0798  2 ! We begin by removing the obsolete PUB from the PUB chain.
;  480         0799  2
;  481         0800  2 remque(.obsolete_pub,p);
;  482         0801  2
;  483         0802  2 ! Now we remove and kill any CTL buffers hanging off the PUB.
;  484         0803  2
;  485         0804  2 while not remque(.p[pub_q_ctlhead0],c) do
;  486         0805  2         phn$kill_ctl(.c);
;  487         0806  2
;  488         0807  2 ! Now we can free up the memory occupied by the PUB.
;  489         0808  2
;  490         0809  2 status = lib$free_vm(p[pub_l_length],p);
;  491         0810  2 check (.status);
;  492         0811  2
;  493         0812  2 return;
;  494         0813  2
;  495         0814  1 end;



                                         0004 00000          .ENTRY  PHN$KILL_PUB, Save R2                            ; 0790
                          7E        04    BC   0F 00002          REMQUE  @OBSOLETE_PUB, P                                 ; 0800
```

```
                   50              6E  D0 00006 1$:     MOVL     P, R0                                                   ; 0804
                   52      0104    D0  0F 00009         REMQUE   @260(R0), C                                             ;
                                   09  1D 0000E         BVS      2$                                                      ;
                                   52  DD 00010         PUSHL    C                                                       ; 0805
             0000V CF              01  FB 00012         CALLS    #1, PHN$KILL_CTL                                        ;
                                   ED  11 00017         BRB      1$                                                      ;
                                   5E  DD 00019 2$:     PUSHL    SP                                                      ; 0809
         7E     04 AE              08  C1 0001B         ADDL3    #8, P, -(SP)                                            ;
        00000000G  00              02  FB 00020         CALLS    #2, LIB$FREE_VM                                         ;
                   09              50  E8 00027         BLBS     STATUS, 3$                                              ; 0810
                                   50  DD 0002A         PUSHL    STATUS                                                  ;
        00000000G  00              01  FB 0002C         CALLS    #1, LIB$SIGNAL                                          ;
                                       04 00033 3$:     RET                                                              ; 0814

; Routine Size:  52 bytes,    Routine Base:  $CODE$ + 02B4
```

```
1 %sbttl 'PHN$MAKE_CTL - Create a Conversation Text Line'
1 !++
1 ! Functional Description:
1 !       This routine is called to allocate and initialize a new Conversation
1 !       Text Line (CTL) buffer.
1 !
1 ! Formal Parameters:
1 !       line_text       Address of descriptor of text to go in CTL (optional).
1 !       ctl_address     Address of longword in which to return CTL address.
1 !
1 ! Implicit Inputs:
1 !       global data
1 !
1 ! Implicit Outputs:
1 !       global data
1 !
1 ! Returned Value:
1 !       none
1 !
1 ! Side Effects:
1 !
1 !--
1
1
2 global routine phn$make_ctl(line_text,ctl_address): novalue = begin
2
2 bind
2       line_text_dsc = .line_text: descriptor;
2
2 own
2       sequence_stamp: long initial(0);
2
2 local
2       status: long,
2       c: ref ctl;
2
2 builtin
2       nullparameter;
2
2
2 ! We begin by allocating memory for the CTL.
2
2 status = lib$get_vm(%ref(ctl_k_size),c);
2 check (.status);
2
2 ! Now we initialize the CTL by filling in its length, the sequence
2 ! stamp (so we can tell the order in which CTLs were created), and the
2 ! line text if specified.
2
2 c[ctl_l_length] = ctl_k_size;
2 c[ctl_l_stamp] = inc (sequence_stamp);
3 begin
3 bind
3       line_dsc = c[ctl_q_line]: descriptor;
3
3 line_dsc[0,0,32,0] = 0;
4 if not nullparameter(1) then (
;  554         0872  4          ch$move(.line_text_dsc[len],.line_text_dsc[ptr],c[ctl_t_linebuf]);
;  555         0873  4          line_dsc[len] = .line_text_dsc[len];
;  556         0874  3 );
;  557         0875  3 line_dsc[ptr] = c[ctl_t_linebuf];
;  558         0876  2 end;
;  559         0877  2
;  560         0878  2 ! Finally, return the CTL address as requested.
;  561         0879  2
;  562         0880  2 .ctl_address = .c;
;  563         0881  2 return;
;  564         0882  2
;  565         0883  1 end;


                                                                         .PSECT  $OWN$,NOEXE,2

                                            00000000  00000 SEQUENCE_STAMP:
                                                                         .LONG   0                                       ;


                                                                         .PSECT  $CODE$,NOWRT,2

                                                   01FC 00000            .ENTRY  PHN$MAKE_CTL, Save R2,R3,R4,R5,R6,R7,R8     ; 0839
                          5E                  08   C2 00002              SUBL2   #8, SP                                  ;
                          58          04      AC   D0 00005              MOVL    LINE_TEXT, R8                           ; 0842
                                      04      AE   9F 00009              PUSHAB  C                                       ; 0857
                    04    AE          67      8F   9A 0000C              MOVZBL  #103, 4(SP)                             ;
                                      04      AE   9F 00011              PUSHAB  4(SP)                                   ;
             00000000G    00                  02   FB 00014              CALLS   #2, LIB$GET_VM                          ;
                          09                  50   E8 0001B              BLBS    STATUS, 1$                              ; 0858
                                              50   DD 0001E              PUSHL   STATUS                                  ;
             00000000G    00                  01   FB 00020              CALLS   #1, LIB$SIGNAL                          ;
                          57          04      AE   D0 00027 1$:          MOVL    C, R7                                   ; 0864
                    08    A7          67      8F   9A 0002B              MOVZBL  #103, 8(R7)                             ;
         50      0000'    CF                  01   C1 00030              ADDL3   #1, SEQUENCE_STAMP, R0                  ; 0865
                 0000'    CF                  50   D0 00036              MOVL    R0, SEQUENCE_STAMP                      ;
                    0C    A7                  50   D0 0003B              MOVL    R0, 12(R7)                              ;
                          56          10      A7   9E 0003F              MOVAB   16(R7), R6                              ; 0868
                                              66   D4 00043              CLRL    (R6)                                    ; 0870
                                              6C   95 00045              TSTB    (AP)                                    ; 0871
                                              0E   13 00047              BEQL    2$                                      ;
                                      04      AC   D5 00049              TSTL    4(AP)                                   ;
                                              09   13 0004C              BEQL    2$                                      ;
      18    A7      04    B8                  68   28 0004E              MOVC3   (R8), @4(R8), 24(R7)                    ; 0872
                          66                  68   B0 00054              MOVW    (R8), (R6)                              ; 0873
                    04    A6          18      A7   9E 00057 2$:          MOVAB   24(R7), 4(R6)                           ; 0875
                    08    BC                  57   D0 0005C              MOVL    R7, @CTL_ADDRESS                        ; 0880
                                                   04 00060              RET                                             ; 0883

; Routine Size:  97 bytes,     Routine Base:  $CODE$ + 02E8
;  567          0884  1 %sbttl 'PHN$KILL_CTL - Deallocate Dead CTL'
;  568          0885  1 !++
;  569          0886  1 ! Functional Description:
;  570          0887  1 !       This routine is called to deallocate an obsolete CTL.  We just
;  571          0888  1 !       free up its memory.
;  572          0889  1 !
;  573          0890  1 ! Formal Parameters:
;  574          0891  1 !       obsolete_ctl    The address of the obsolete CTL.
;  575          0892  1 !
;  576          0893  1 ! Implicit Inputs:
;  577          0894  1 !       global data
;  578          0895  1 !
;  579          0896  1 ! Implicit Outputs:
;  580          0897  1 !       global data
;  581          0898  1 !
;  582          0899  1 ! Returned Value:
;  583          0900  1 !       none
;  584          0901  1 !
;  585          0902  1 ! Side Effects:
;  586          0903  1 !
;  587          0904  1 !--
;  588          0905  1
;  589          0906  1
;  590          0907  2 global routine phn$kill_ctl(obsolete_ctl): novalue = begin
;  591          0908  2
;  592          0909  2 bind
;  593          0910  2         c = .obsolete_ctl: ctl;
;  594          0911  2
;  595          0912  2 local
;  596          0913  2         status: long;
;  597          0914  2
;  598          0915  2
;  599          0916  2 status = lib$free_vm(c[ctl_l_length],obsolete_ctl);
;  600          0917  2 check (.status);
;  601          0918  2 return;
;  602          0919  2
;  603          0920  1 end;



                                          0000 00000          .ENTRY  PHN$KILL_CTL, Save nothing                    ; 0907
                                   04   AC  9F 00002          PUSHAB  OBSOLETE_CTL                                  ; 0916
                  7E       04   AC          08  C1 00005          ADDL3   #8, OBSOLETE_CTL, -(SP)                       ;
                     00000000G  00          02  FB 0000A          CALLS   #2, LIB$FREE_VM                               ;
                                09          50  E8 00011          BLBS    STATUS, 1$                                    ; 0917
                                            50  DD 00014          PUSHL   STATUS                                        ;
                     00000000G  00          01  FB 00016          CALLS   #1, LIB$SIGNAL                                ;
                                                04 0001D 1$:      RET                                                   ; 0920

; Routine Size:  30 bytes,    Routine Base:  $CODE$ + 0349


;  604          0921  1
;  605          0922  0 end eludom
```

```
                                                                              .EXTRN  LIB$SIGNAL

;                                     PSECT SUMMARY
;
;       Name                     Bytes                          Attributes
;
;   $CODE$                          871  NOVEC,NOWRT,  RD ,  EXE,NOSHR,  LCL,  REL,  CON,NOPIC,ALIGN(2)
;   $PLIT$                            4  NOVEC,NOWRT,  RD ,NOEXE,NOSHR,  LCL,  REL,  CON,NOPIC,ALIGN(2)
;   $OWN$                             4  NOVEC,  WRT,  RD ,NOEXE,NOSHR,  LCL,  REL,  CON,NOPIC,ALIGN(2)



;                              Library Statistics
;
;                                                -------- Symbols --------          Pages          Processing
;       File                                     Total    Loaded   Percent          Mapped         Time
;
;   _$255$DUA28:[SYSLIB]STARLET.L32;1             9776         0         0             581            00:00.8



;                                      COMMAND QUALIFIERS

;       BLISS/CHECK=(FIELD,INITIAL,OPTIMIZE)/LIS=LIS$:PUBSUBS/OBJ=OBJ$:PUBSUBS MSRC$:PUBSUBS/UPDATE=(ENH$:PUBSUBS)

; Size:            871 code + 8 data bytes
; Run Time:           00:13.3
; Elapsed Time:       00:52.7
; Lines/CPU Min:        4162
; Lexemes/CPU-Min: 32641
; Memory Used:  175 pages
; Compilation Complete
```

0305 AH-BT13A-SE
VAX/VMS V4.0

DIGITAL EQUIPMENT CORPORATION
CONFIDENTIAL AND PROPRIETARY